第 1 章　绪　　论

计算机系统在国民经济和人们生活中起着越来越重要的作用。操作系统是计算机系统不可或缺的核心系统软件，是计算机系统的调度、控制中心。一方面，操作系统将裸机改造成为功能强大、各部件高效运行、使用方便灵活、安全可靠的虚拟机，为用户提供计算机系统的良好使用环境；另一方面，操作系统采用合理有效的方法组织多个用户任务共享计算机的各种资源，最大限度地提高资源的利用率。

自世界上第一台计算机 ENIAC 于 1946 年问世以来，计算机在运算速度、存储容量、元器件工艺及系统结构等方面都有了惊人的发展。以前，人们按照计算机元器件工艺的演变过程将计算机的发展划分为 4 个时代：电子管时代、晶体管时代、集成电路时代和大规模集成电路时代。与硬件发展相类似，人们也将操作系统的演变和发展过程划分为 4 个时代：单道批处理时代，多道批处理、分时和实时系统时代，同时具有多方面功能的多方式系统时代，并行与分布式系统时代。

本章将介绍什么是操作系统及操作系统在计算机系统中的地位和作用。并通过阐述操作系统历史的演变过程，使读者对操作系统的基本概念及技术的产生和发展等问题有一个直观的了解，从而使读者对不同类型操作系统的基本特征、今后的发展方向及目前流行的操作系统有更深刻的认识。

1.1　什么是操作系统

众所周知，处理机（CPU）、主存、辅存、终端、网络设备等硬件资源通过主板连接构成了看得见、摸得着的计算机硬件系统。为了能使这些硬件资源高效地、尽可能并行地被用户程序所使用，也为了给用户程序提供易用的访问这些硬件的方法，我们必须为计算机配备操作系统软件。操作系统的工作就是管理计算机的处理机、主存、外部设备等硬件资源，提供存放于存储设备中的文件等逻辑资源，并组织用户任务（如以进程的形式）使用这些资源。

操作系统是一种系统软件，是软、硬件资源的控制中心，它以尽量合理有效的方法组织单个或多个用户以多任务方式共享计算机系统的各种资源。

1.1.1　计算机系统的软件构成

计算机系统的软件层次及构成如图 1.1 所示。

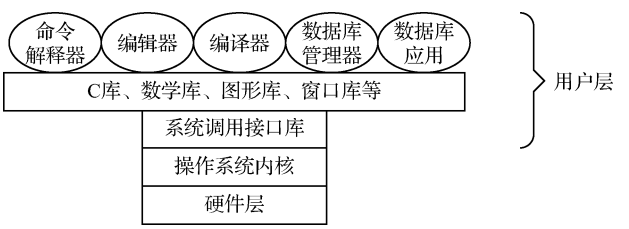

图 1.1　计算机系统的软件层次及构成

当用户在计算机中安装操作系统时，如图 1.1 所示的操作系统内核、命令解释器、编辑器、编译器，以及各种库程序，甚至数据库管理器、数据库应用等都从安装介质复制到计算机系统的磁盘中。

无论是安装 UNIX、Linux 还是 Windows 操作系统，命令解释器都是一个必不可少的程序。用户通过它来使用计算机。如果没有它，用户就无法操作计算机，无法输入命令让计算机去执行（注意，在窗口界面中，用户通过与命令解释器对应的程序管理器，如 Windows 的 explorer.exe，来使用计算机）。在现代操作系统实现中，命令解释器不是操作系统内核的组成部分，但它在使用计算机的过程中是不可缺少的。用户在终端输入的命令就是由命令解释器接收并解释执行的。其他的操作系统内核层之上的程序则是根据计算机的定位（服务器或工作站）而选择安装的。如果将计算机定位成开发程序用的工作站，那么用户必须安装编辑器进行程序编辑，并安装编译器进行程序编译。如果把计算机作为一个网络上的 Web 服务器，那么必须安装 Web 服务器程序。无论是用户自编的普通 C 语言程序还是数据库应用程序，都需要在安装并运行操作系统后再进行开发或安装。这些在操作系统内核层之上的程序，不管是命令解释器、Web 服务器，还是用户自编的程序，都要通过操作系统提供的进程机制来运行。

从狭义上看，操作系统只包含如图 1.1 所示的**操作系统内核**。它是一个非常重要的系统程序，管理着系统中所有的公共资源，并提供实现程序运行的进程机制。由于操作系统内核工作的重要性、特殊性，它必须在一种特殊的保护状态下运行，以免受到用户层程序的干扰和破坏，因此，它提供一组称为**系统调用**的接口，供上层程序调用，从而保证操作系统内核在特殊保护状态下运行，并且满足上层程序对系统资源的申请、使用、释放，以及进程的创建、结束等诸多功能的调用。

图 1.1 中的各种库程序实际上就是一些可以重用的、公用的子程序，它们提供形形色色的功能。系统提供这些库程序是为了方便用户编程。用户不必为了实现一个通用的功能再重写库程序代码，而只要引用库程序中的函数即可。库程序可以看成一些通用的、公共的程序集合，利用操作系统内核提供的简单的资源管理功能实现复杂的复合功能。这些通用的公共程序之所以不放到操作系统内核中实现，是因为它们不涉及系统公共资源的管理，也是为了控制操作系统内核的大小。

1.1.2 操作系统作为特殊子程序

从图 1.1 可以看出，操作系统内核位于计算机硬件之上。操作系统内核为用户层程序提供系统调用功能。系统调用与普通函数调用相似，可以看成对特殊的公共子程序的调用。因为这些程序提供了一些可以被任意用户层程序调用的公共功能，所以用户不需要再编写实现这些功能的程序，只要调用操作系统内核提供的相应“系统调用”即可。但是，要特别注意系统调用的特殊性，即系统调用处理程序运行在一种特殊的保护状态下。在这种状态下，程序可以执行一些特权指令，访问用户层程序访问不到的系统存储空间。系统调用之所以具有这样的特殊性，是因为系统调用处理程序涉及系统共享资源的操作。

举例来说，求 $\sqrt{x}$ 的值是许多用户程序都要做的工作，可以把它作为一个公共子程序实现。那么它需要作为系统调用在操作系统内核中实现吗？回答是否定的。虽然计算 $\sqrt{x}$ 需要许多条机器指令来实现，但是因为它不涉及系统的共享资源，只对输入变量 x 进行操作，所以可以把它作为数学库中的子程序来实现。

计算机都使用磁盘来存放系统或用户的程序及数据，存放在磁盘中的程序或数据称为**文件**。当一个用户程序要访问某个文件中的某段信息的时候，需要知道这段信息存放在磁盘哪个扇区中，需要向磁盘控制器发送读扇区的请求，需要查看扇区信息是否已经读入主存。如果这些操

作都交给用户来编程实现，不仅复杂，而且重复。因此，操作系统给用户提供一个简单的、统一的文件操作界面，即磁盘中可以包含许多文件，每个文件可以按照读/写方式打开，然后进行读/写操作，最后关闭文件。用户无须知道磁盘是怎么工作的、如何读/写数据，也不需要知道要读/写的数据放在磁盘的哪个扇区中，只需要知道读/写哪个文件的哪一段数据即可。利用这个简单的文件操作界面就可以与磁盘进行数据交换。至于确定文件信息在哪个扇区中、如何读/写这个扇区，则由操作系统的系统调用处理程序来实现。因为磁盘不是某个用户的私有资源，磁盘中的文件可以供多个用户访问，涉及磁盘和文件的管理数据都应该受到保护，所以文件操作以操作系统内核系统调用的形式实现。

1.1.3 操作系统作为资源管理者

计算机由处理机、主存、辅存、终端、网络设备等硬件资源组成。处理机提供程序执行能力；主存、辅存提供程序和数据的存储能力；终端提供人机交互能力；网络设备提供计算机间通信能力。这些硬件资源要能被计算机用户高效地使用，必须有适合每种硬件资源特点的资源分配和使用机制。

为使硬件资源充分发挥作用，必须允许多用户或单用户以多任务方式同时使用计算机，以便让不同的资源由不同的用户任务同时使用，减少资源的闲置时间。例如，当一个用户任务将文件内容从辅存向主存的缓冲区读出时，另一个用户任务可以让自己的程序在处理机上运行。这样，处理机、主存、辅存同时工作，也就提高了资源利用率。

要让每种资源被多用户任务充分利用，就需要研究每种资源的特点。对于单处理机来说，它只能执行一个指令流。如果多个用户任务都要使用它，那只有让多个用户任务的程序分时地在处理机上运行，也就是说，处理机交替地运行多个用户任务中的程序。这意味着，操作系统要合理调度多用户任务来使用处理机。存储设备为程序和数据提供存储空间，只要多个用户的程序和数据按照规定的位置存放，互不交叉占用，它们就可以共存。操作系统要做的事就是管理存储空间，把适用的空间分配给用户的程序和数据使用；当用户任务访问这些程序和数据时要能够找到它们。

针对不同资源的特点，资源管理提供两种资源共享使用的方法：“时分”和“空分”。

① **时分**是指由多个用户进程分时地使用该资源。除上述的处理机外，还有很多其他的资源也必须分时地使用，如外部设备控制器、网卡等，这些控制部件包含了控制 I/O 的逻辑，必须分时地使用。

② **空分**是针对存储资源而言的。存储资源的空间可以被多个用户进程共同以分割的方式占用。

在时分共享使用的资源当中，有如下两种不同的使用方法。

① **独占式**使用。独占表示某个用户任务占用该资源后，可以执行对资源的多个操作，使用一个完整的周期。例如，如果多个用户任务使用打印机，那么对打印机的独占式使用是指多个用户任务一定是分时地使用该打印机的；每个用户任务使用打印机时，执行了多条打印指令，打印了一个完整的对象（如完整的文件）。这里，每个用户任务要执行多条打印指令，为了不让多条打印指令在执行过程中被别的打印任务中断，用户任务需要在执行打印指令前申请独占该打印机资源，执行完所有打印指令后再释放。

② **分时式**共享使用。这种共享使用是指用户任务占用该资源后无须使用一个逻辑上的完整周期。例如，对处理机的占用，用户任务随时都可以被剥夺处理机，只要运行现场保存好了，下次该用户任务再次占用处理机时就可以继续运行。再如，对磁盘的输入/输出，当一个用户任务让磁盘执行一个 I/O 请求后，其他用户任务也可向磁盘发出 I/O 请求，操作系统并不要求某个

用户任务的几个 I/O 请求之间不能插入其他用户任务的 I/O 请求。

操作系统应针对不同的资源类型，实现不同的资源分配和使用策略，并为资源分配、释放、使用提供相应的系统调用接口。

1.1.4 操作系统提供程序并发运行机制

用户可使用计算机进行科学计算、数据管理、通信、控制等工作。要实现所述的这些任务，必须执行相应的程序。用户使用处理机来执行程序，用程序驱动外部设备来进行数据交换，驱动网络设备来进行通信。用户的意图必须由程序及程序的输入参数表示出来。为了实现用户意图，必须让实现相应功能的程序得以执行；为了能让程序得以执行，需要由操作系统给程序及程序数据安排存放空间；为了能提高资源利用率，增加并发度，还必须让多个用户程序能分时占用处理机；为了能让一个程序还没运行完就允许另一个程序占用处理机运行，就必须保存上一个程序的运行现场。因此，必须要对实现各种用户意图的各个程序的执行过程进行描述和控制。

说明程序执行的状态、现场、标识等各种信息，有选择地调度某个程序占用处理机运行，这些工作必须由操作系统完成，这也是为了实现程序对处理机的分时占用。

操作系统一般用**进程**机制来实现程序的执行。

进程是指程序的执行，即程序针对某个数据集合的执行过程。操作系统的进程调度程序决定处理机如何在各执行程序间的切换。操作系统为用户提供进程创建和结束等的系统调用功能，使用户能够创建新进程以运行新的程序。

操作系统在系统初始化后，会为每个可能的系统用户创建第一个用户进程，用户的其他进程则可以由先前生成的用户进程通过“进程创建”系统调用陆续创建，以完成用户的各种任务。

在支持交互使用计算机的系统中，用户的第一个进程往往需要运行命令解释器程序（对于图形窗口终端用户而言，就是具有窗口界面的程序管理器，如 Windows 操作系统的 explorer.exe），这个程序会从终端获得用户输入的命令（或用户单击执行程序图标的信息），再进行相应的处理，可能会调用操作系统的“创建进程”系统调用，创建新进程去运行实现命令功能的程序。例如，在 Linux 操作系统控制的终端上输入：

```
$ cp /home/ly/test.c /home/wq/hello.c
```

那么，这一行字符串会由命令解释器程序获得，它会创建一个子进程，由子进程去运行 cp 实用程序，由 cp 实用程序建立一个新文件/home/wq/hello.c，并把/home/ly/test.c 文件中的内容读出来，写到/home/wq/hello.c 中。

1.2 操作系统的发展历史

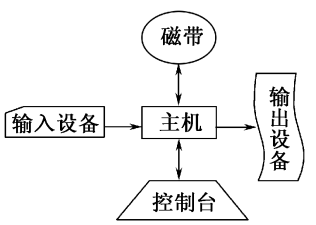

图 1.2 早期计算机系统

在计算机刚刚诞生的 20 世纪 40 年代，计算机系统仅由硬件和应用软件组成。在这一时期，整个计算机系统是由用户直接控制使用的，所以又称为“手工操作”阶段。当时的计算机不仅速度慢、存储容量小，而且外部设备（简称外设）简单，主要使用磁带作为辅存，如图 1.2 所示。整个计算机系统由单个用户独占使用。当时用户使用计算机的大致方法是：将程序和数据以穿孔方式记录在卡片或纸带上，把卡片或纸带装在输入设备上；然后在控制台上形成输入命令，启动设备将卡片、纸带信息或磁带上的信息输入指定的主存单元；接着在控制台上指定主存启动地址，并启动程序运行；最后在打印机等输出设备上取得程序运行的结果。

显然，在这种使用方式下，用户在上机时独占全部资源，使用机器语言编写程序，并且对计算机各部分的工作直接实施人工干预，或者由用户自己所写的程序控制。在硬件各部分速度较低且程序较小的情况下，这种方式还能被人们所接受。但是，随着计算机速度的提高，以及FORTRAN，COBOL 等高级程序设计语言的问世，这种方式势必使人无法忍受。

例如，用户要运行一个用 FORTRAN 语言编写的程序，必须先把存有 FORTRAN 编译程序的磁带安装在磁带机上，将 FORTRAN 编译程序和用户编写的 FORTRAN 源程序调入主存，并对 FORTRAN 源程序进行编译；然后再安装磁带上的“链接程序”，对编译好的程序进行链接，形成目标程序；最后启动目标程序运行。

由此可见，一批包括语言编译程序在内的系统软件的问世，使用户上机过程变得更繁杂，并增加了程序运行前的准备时间。由于计算机运算速度的提高，因此上述人工操作势必造成更大的资源浪费。为了缩短运行前的准备时间，提高计算机资源的利用率，人们提出了简单的改进措施，引入了“系统操作员”的概念。各用户将自己的程序及程序的运行步骤（控制意图）交给系统操作员，系统操作员将这种形式的一批用户作业按类进行划分，每次处理一类作业。

例如，将需要进行 FORTRAN 编译程序的作业组织成一类依次进行编译，并由系统操作员控制计算机运行用户程序。当然，这种使用计算机的方法仍旧停留在手工操作阶段，人的操作速度与机器的运行速度相比，仍存在极大的差距。由于人的操作缓慢，使得计算机资源在大部分时间被闲置，因此急需用程序来代替人的手工操作。

1.2.1 监督程序

20 世纪 50 年代，为了减少系统操作员工作所花的时间，提高资源利用率，人们开始利用计算机系统中的软件来代替系统操作员的部分工作，从而产生了最早的操作系统——早期批处理系统。

1．批处理系统

批处理系统的基本思想是：设计一个常驻主存的程序（监督程序，Monitor）。由系统操作员有选择地把若干用户作业合成一批，安装在输入设备中，并启动监督程序。然后，由监督程序自动控制这批作业运行。监督程序首先把第一道作业调入主存，并启动该作业；一道作业运行结束后，再把下一道作业调入主存启动运行。待一批作业全部处理结束后，系统操作员则把作业运行的结果一起交给用户。按照这种方式处理作业，各作业间的转换及各作业的运行完全由监督程序自动控制，从而减少了部分人工干预，有效地缩短了作业运行前的准备时间。

所谓**作业**（Job），是用户在一次上机活动中要求计算机系统所做的一系列工作的集合。从执行的角度看，作业由一组有序的**作业步**组成，如编译、运行分别称为不同的作业步。

当监督程序取代系统操作员的部分工作后，用户应以某种方式告知监督程序其作业的处理步骤。因此，在早期批处理系统中引出了“作业控制语言”和“作业控制说明书”的概念。作业控制说明书是使用作业控制语言编写的、用于控制作业运行的一段描述程序。在组织一道作业时，通常将作业控制说明书放在被处理的作业前面（或插入适当位置），由监督程序通过解释执行作业控制说明书中的语句来控制作业的运行。典型的卡片作业结构如图 1.3 所示。

这叠卡片中某些卡片表示了作业控制说明书中的语句。监督程序通过逐条解释、执行该作业控制说明书中的语句来自动控制作业运行，具体步骤说明如下。$JOB 语句说明该作业的名字、预计最大执行时间等信息。监督程序解释$FORTRAN 语句的结果是，把 FORTRAN 编译程序调入主存，并启动其编译后面的源程序。编译结束后，控制返回给监督程序。监督程序解释$LOAD 语句的结果是，通过链接程序把经过编译的程序链接起来，形成可执行程序。最后，解释$RUN 语句，从而启动可执行程序运行。

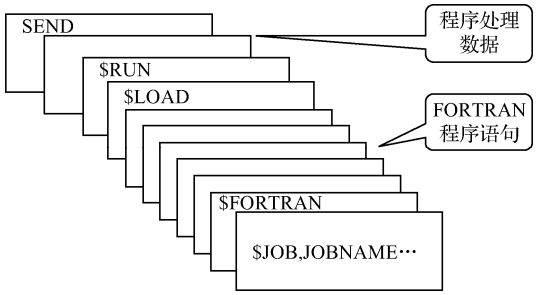

图 1.3　典型的卡片作业结构

监督程序内专设一个作业控制程序（Job-Controller）以控制作业的运行。作业的控制意图描述在作业控制说明书中。作业控制程序在控制某道作业运行时，其实质性工作是解释执行作业控制说明书中的语句，实现对作业的控制。从逻辑上看，作业由三部分组成：源程序（或程序）、数据及“加工”步骤。监督程序一旦接收到一道作业，就开始根据“加工”步骤所规定的动作逐步完成对作业的加工活动。

如果用户可以使用全部的机器指令，就可以直接控制和使用系统资源（如主存、外部设备等），但是用户编程中的错误往往可能导致各种预想不到的后果。为了避免这类错误发生，人们将机器指令分为“普通指令”和“特权指令”，并且引入了“**模式/态**”（Mode）的概念。把有关 I/O 的指令、对特殊寄存器的访问等列为特权指令，并且规定只有监督程序才有权执行特权指令，用户程序只能执行普通指令。将 I/O 指令列为特权指令后，用户便不能直接控制设备进行传输了。如果用户希望进行 I/O 操作，则必须向监督程序提出请求；监督程序通过调用系统内部的程序段来完成用户的 I/O 请求。由此引出了“**系统调用**”（System Call）或称“广义指令”的概念。

监督程序为用户提供一系列分别完成各种不同功能的系统调用程序段。用户程序中可以用一条特殊的硬件转移指令（不带转移地址，直接转到操作系统与硬件约定的地址）请求一次特定的（通过编号区分）系统调用。当处理机执行到用户程序中的系统调用指令时，硬件通过产生“**自陷**”（Trap）并借助转换机制将当前的用户模式转变为监督模式，随之转入监督程序运行。监督程序根据用户提供的调用参数进行相应的处理，如完成 I/O 操作等功能。处理结束后，监督程序根据“自陷”前所保存的现场将模式改变为用户模式，退回用户程序继续执行。

“系统调用”概念的引入提高了监督程序在整个系统中的地位，丰富了监督程序的功能。监督程序不仅对作业的处理流程进行自动控制，而且还负责为用户程序的运行提供各种功能的服务。系统调用的引入也为用户提供了使用计算机系统的新接口，使用户从直接使用物理处理机的繁杂束缚中解脱出来，呈现在用户面前的是一台功能强大、使用方便的虚拟处理机。引入系统调用后，用户对系统内部各种资源的使用均由监督程序代为完成，因而也使系统更加安全，既避免了用户在编程使用资源时可能出现的某些错误，也有利于提高资源利用率。

在手工操作阶段，存储器全部由用户支配使用。引入监督程序后，存储器不再由用户独占，常驻主存的监督程序必须占用部分主存空间。通常，监督程序占用主存的 0～k 单元，而 k+1～n 单元由用户程序占用。监督程序所在的存储空间称为“**系统空间**”，用户程序所在的存储空间称为“**用户空间**”。为了避免用户程序执行时有意或无意地对系统空间进行存取访问，在硬件方面提供一个界地址寄存器，用于存放系统空间与用户空间的分界地址。若系统处于用户模式，每次访问主存时，硬件会自动进行地址越界检查，从而保证了监督程序不被破坏。这种保护称为“**存储保护**”。

在早期批处理系统中，当系统动态运行时，可能一段时期处于监督模式，一段时期又处于用户模式。从用户模式进入监督模式主要是由用户程序中的系统调用引起的。例如，用户请求

设备 I/O 或请求结束运行。但是，若在用户程序执行过程中永远不出现系统调用，或者永远不出现请求结束运行的系统调用（如用户程序进入了“死循环”），监督程序便失去了作用。为了防止这种情况发生，人们设置了“**定时器中断**”。

定时器（Timer）是一个硬件计数器，其计时长度可以根据需要而调整。计数器根据硬件的计时周期自动计时，计数器满后便发生定时器中断。用户程序执行时，若碰到定时器中断，则无条件进入监督程序。监督程序根据当前作业控制说明（或规定）的“最大运行时间”值来判断该程序是否进入了“死循环”，从而可以有效地防止某个用户程序长期垄断处理机的现象。

引出上述概念后，早期批处理系统中的监督程序工作流程如下。

（1）判断输入设备中是否有待输入的作业，如果没有，则等待作业输入。

（2）从输入设备输入一道作业。

（3）控制作业运行。

① 取一条作业控制说明书中的语句，解释执行。如果是一条“作业终止”语句，则删除该作业，转第（1）步。

② 如果当前是一条“执行性语句”（如请求编译、请求运行用户程序等），则在主存中建立相应程序的运行环境，并分配处理机，开始在用户模式下执行该程序。

③ 在用户模式程序的执行过程中，如果发生“中断”事件（如 I/O 中断、系统调用、程序执行错误等），将控制转入监督程序。当“中断”事件处理结束后，返回用户态，用户程序继续执行。

④ 用户程序执行结束后，进入监督程序，控制转第①步，取下一条作业控制说明书中的语句执行。

监督程序如同一个系统操作员，它负责批处理作业（即按照作业控制说明书运行的作业）的 I/O，并自动根据作业控制说明书，以单道串行的方式控制作业运行，同时在程序运行过程中通过提供各种系统调用的方式来控制计算机资源的使用。虽然监督程序并不能被称为操作系统（它与操作系统的本质差别在于监督程序不具有并发控制机制），但监督程序在系统中的地位和作用、实现的基本目标及管理资源的基本方法与操作系统类似。真正的操作系统就是在此基础上进一步发展和完善起来的。

与手工操作阶段相比，监督程序的引入有效地减少了人工干预时间和作业运行前的准备时间，相对提高了处理机的利用率。但是，在计算机运算速度大幅度提高的形势下，用这种方法管理计算机远不能满足需要。首先，在一个处理机上运行的程序启动 I/O 操作时，处理机被迫处于空闲状态或**忙等待**（Busy_Wait）状态，也就是说，处理机启动 I/O 操作后，再循环判断 I/O 操作是否完成，而没有做实质性工作，这将导致高速的处理机受到慢速外部设备的牵制，从而使处理机无法充分得到利用。

2．利用脱机 I/O 技术改善系统性能

因为作业的 I/O 操作与作业的运行是串行的，所以受卡片机、光电机、打印机等慢速 I/O 设备的影响，处理机的利用率难以提高。为了进一步提高系统的工作效率，必须解决低速 I/O 的问题。磁带机的传输速度比卡片机、光电机和打印机的速度快，若用磁带机代替这类低速设备便可进一步缩小处理机与外部设备之间在速度上的差异。历史上，人们曾采用脱机 I/O 技术实现作业的 I/O 操作，其系统模型如图 1.4 所示。

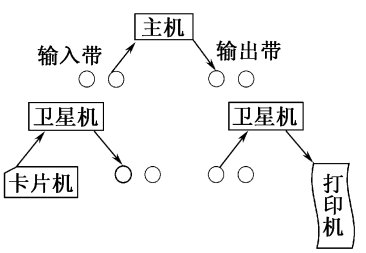

图 1.4　脱机 I/O 技术系统模型

在采用脱机 I/O 技术的系统中，主机的所有 I/O 都是通过磁带机进行的。用户的作业由另一台能力较弱、价格较低的卫星机负责从卡片机传输到磁带（称为输入带）上，然后系统操作员将输入带安装到与主机相连的磁带机上。主机在处理输入带上的作业时，将产生的输出结果直接送到输出带上。系统操作员再将输出带安装到卫星机上，由卫星机负责将输出带上的信息从打印机上输出。由于磁带机的速度比慢速 I/O 设备（如卡片机、打印机）的快，因而按照这种脱机方式控制作业的 I/O 操作，可以减少其所花费的时间，有效地提高处理机的利用率。如果将一台主机与多台卫星机有机地组合，使速度得到最好的匹配，则可以大幅度提高系统的处理能力。从 20 世纪 50 年代末到 60 年代初，这种脱机处理方式被广泛地应用于批处理系统中。

无论如何，由于处理机与 I/O 设备是以串行方式工作的，也就是说，当处理机工作时，I/O 设备闲着；当 I/O 设备工作时，处理机处于忙等待，这就限制了设备的利用率。另外，从方便用户的角度来说，采用这种批量处理的控制方法，用户不能以交互方式使用计算机，从而限制了对计算机的灵活使用。随着对这些问题不断深入的研究和解决，逐步形成了第 2 代操作系统。

1.2.2 专用操作系统

20 世纪 60 年代初，计算机硬件有了很大的发展。例如，主要元器件由电子管变成了晶体管，出现了磁盘、通道、终端等部件。而这些硬件的发展为监督程序提出了新的研究课题，也为操作系统的形成提供了重要的物质基础。这一时期是操作系统形成的重要时期。随着计算机应用的巨大牵引，不仅批处理系统得到了充分的发展，而且还出现了实时（Real Time）、分时（Sharing Time）等不同类型的系统。

1．多道批处理系统

在早期批处理系统（也称单道批处理系统）中，处理机与 I/O 设备以串行方式工作，故两者的利用率较低。为了提高资源利用率，人们开始使用 I/O 缓冲、SPOOLing 等技术，尤其是引入了“多道程序设计”（Multiprogramming）的思想，使单道批处理系统发展为多道批处理系统。

（1）利用 I/O 缓冲实现异步编程

在单道批处理系统中，作业的处理过程是单道串行的，所以在监督程序的控制下，处理机与外部设备也按串行方式工作。为改变这种串行工作方式，人们首先采用了缓冲（Buffering）技术使两者在一定程度上并行操作。

例如，在主存中建立两个长度相同的缓冲区：B0，B1。对于一批待输入的信息，首先将其中的一个记录从外部设备中读入 B0，读完后接着将下一个记录从外部设备中读入 B1，与此同时，处理机开始处理 B0 中的记录。待处理机的处理工作与输入工作均结束后，则将下一个记录读入 B0，处理机同时处理 B1 中的记录。如此重复，直至将信息全部输入并处理完毕，这种利用双缓冲区实现的 I/O 操作在一定程度上实现了处理机与外部设备并行工作。这类并行的实现要求 I/O 设备有较强的功能，能够不依赖于处理机实现外部设备与主存独立地交换数据，在引入通道技术后这个缓冲技术得到了广泛应用。

（2）SPOOLing 技术

通道技术的引入，使处理机与外部设备并行操作成为可能。

通道是指专门用来控制 I/O 的硬件装置，可以实现外部设备与主存直接交换数据，在相当长的时间里不用打扰处理机，因此这时处理机可以去干别的事情。为了能够消除脱机 I/O 带来的人工干预的麻烦，又要保持脱机 I/O 系统中作业高速入/出主存的特点，人们借助通道和磁盘成功地实现了著名的 SPOOLing 系统。通道也可看成专门的 I/O 处理机，磁盘则是一种比磁带更快

且能够随机存取的辅存。

SPOOLing（Simultaneous Peripheral Operation On Line 的简称）的含义是并行的外部设备联机操作。用 SPOOLing 技术控制批处理系统中作业 I/O（如图 1.5 所示）的基本思想是：用磁盘（或一组磁盘）设备作为主机的直接 I/O 设备，即系统直接从磁盘中选取作业运行，作业的运行结果也直接存入磁盘，相应的通道（在设备驱动程序驱动下）则负责并行地将卡片机中的用户作业输入磁盘，或者将磁盘中作业的运行结果从打印机上输出。

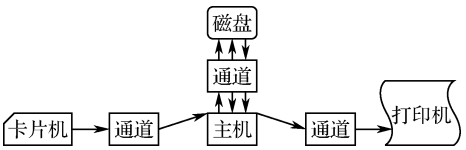

图 1.5　SPOOLing 技术控制批处理系统中作业的 I/O

通道直接受主机控制，主机与通道之间借助**中断机制**相互通信。例如，只要卡片机上有用户作业，系统便启动设备通道，通道被启动后便将作业依次通过主存传输至磁盘中，在作业输入期间，主机可以并行地从事其他工作。类似地，只要磁盘中存在等待输出的信息且打印机空闲，则系统通过启动通道将信息从打印机上输出。所以，SPOOLing 技术又被称为“伪脱机 I/O”技术，被广泛地用于后来的批处理系统中。采用 SPOOLing 技术实现 I/O 与主机并行的系统通常又简称为 SPOOLing 系统。SPOOLing 技术为实现多道批处理系统中的多道程序设计思想提供了重要的基础。

（3）多道程序设计技术

如前所述，人们采用 SPOOLing 技术，利用主机和通道间的并行性，可以使作业的 I/O 与主机运算并行，提高了系统效率。当在主机中运行的某道作业需要传输大量数据时，人们可以采用缓冲技术来获得一定程度上的并行。尽管如此，由于系统中作业之间仍以串行方式被处理，即主存在任何时刻至多保持一道作业，处理完一道作业后再从外部选取另一道作业，因此无法进一步提高资源（如处理机、主存等）的利用率。为了从根本上解决这一问题，人们提出了“多道程序设计”技术。

多道程序设计技术的基本思想是，在主存同时保持多个程序（作业），主机（书中若没有特殊说明，则都是单处理机系统）以交替方式同时处理多个程序。

所谓多个程序，从宏观上看，是指主机内同时保持和处理若干已开始运行但尚未结束的程序。而采用这种多道程序设计技术的系统被称为多道程序设计系统。

由于任何一道作业的运行总是交替地串行使用处理机、外部设备等资源，即：使用一段时间的处理机，然后使用一段时间的 I/O 设备，如图 1.6 所示，因此，如果采用多道程序设计技术，加之对多个程序实施合理的运行调度，则可以大大提高处理机与外部设备的利用率，使两者高度并行工作。

如图 1.7 所示，是三道作业同时运行时处理机与外部设备的利用情况。当作业 A 因请求 I/O 设备而放弃处理机时，操作系统为作业 A 启动相应通道（由通道独立地控制 I/O 设备）后，便把处理机重新分配给作业 B，此时处理机与 I/O 设备并行工作。类似地，当作业 B 也请求 I/O 设备时，处理机重新分配给作业 C。因此，只要系统能保持足够道数的作业，再加上合理的调度，便可能使处理机与 I/O 设备获得高度的并行。采用多道程序设计技术无疑大大提高了主存及其他资源的利用率。

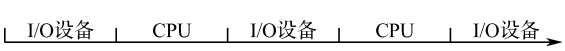

图 1.6　单道作业的执行过程

图 1.7　多道作业的执行过程

引入多道程序设计技术，无论对于实现多道批处理系统，还是交互式、分时系统，乃至实时系统，均提供了重要的技术手段。多道程序设计系统的出现标志着操作系统的形成。

操作系统的最基本特征如下：

① 并发（Concurrent）机制，支持多道程序设计技术。

② 共享（Sharing）机制，控制各种并发活动正确（空分或时分）地共享系统软、硬件资源。

正是由于这两个特征使得操作系统变得极其复杂。

利用上述介绍的各种技术，特别是多道程序设计技术，便将早期的单道批处理系统发展成了多道批处理系统。多道批处理系统的基本特征是：系统按照成批（或称批量）的形式输入用户作业，并采用 SPOOLing 技术和多道程序设计技术控制多道作业运行。

2．分时系统

多道批处理系统的出现有效地提高了系统资源的利用率，但却失去了手工操作阶段的“交互性”优点。也就是说，使用多道批处理系统的用户必须将其作业的控制意图完全地描述在作业控制说明书中。用户一旦把作业交给了系统，便不能再以“会话”方式控制作业运行了，所以用户会在一定程度上感觉不便。

首先，用户的算题周期延长了，用户向机房提交的作业往往需要几经反复才能获得所需结果（对一个新程序尤其如此）；其次，在程序运行过程中失去了人的主观能动作用。

按照手工操作方式算题，程序员可以观察程序的运行情况，一旦发现错误便可以随时设法改正。特别是对一个给定的数学模型，当需要观察不同参数对其产生的影响时，尤其需要交互式环境。“方便用户”是操作系统追求的重要目标之一，所以在这一阶段很快出现了以多道程序设计技术为基础的交互式系统，即“分时系统”。

由于控制台和打印机这类外部设备作为交互作用的人机接口设备极不方便，因此当时作为理想的人机接口设备的终端便应运而生。

终端是集 I/O 能力为一体的设备。在此设备基础上，系统为用户提供一组终端命令，在操作系统中设立一个**命令解释程序**（作用相当于批处理系统中的作业控制程序）。用户可以在终端上通过命令与系统交互，从而产生了交互式系统。将交互式系统与多道程序设计系统相结合便形成了分时系统。

在分时系统中，一台计算机与多台终端相连接。用户通过各自的终端和终端命令以交互方式使用计算机。系统使每个用户都能感觉到好像自己在独占计算机。而在系统内部，操作系统负责协调多个用户分时共享处理机，这便是所谓“分时”的含义。

在协调用户分享处理机时，操作系统通常采用“时间片轮转”原则分配处理机给用户。操作系统规定一个被称为“时间片”的时间单位，所有终端用户轮流享用一个时间片的处理机时间。例如，若有 n 个用户，时间片值为 Q，则每个用户在 nQ 的时间内至少能使用 Q 个时间单位的处理机。由于处理机的速度比人在终端上输入命令的速度快很多，因此用户似乎感到处理机被自己所独占。

分时系统的基本特点如下：

① 并发性。系统能协调多个终端用户同时使用 CPU（即系统内部具有并发机制），能控制多个程序同时运行。

② 共享性。对资源而言，系统在宏观上使各终端用户共享计算机的各种资源，而在微观上它们则分时使用这些资源。

③ 交互性。对系统和用户双方而言，人与系统以对话方式进行工作。

④ 独立性。对用户而言，感觉只有他自己在使用计算机。

显然，前两个特点（即并发和共享）是各类操作系统所共有的基本特征，而后两个特点是分时系统所独有的特点。

交互式作业是指在分时系统中，用户通过终端和系统提供的终端命令指导上机的过程。用户的一次交互使用机器的过程被看成交互式作业。交互式作业的提交形式与批处理作业不同。用户登录代表了用户交互作业的生成，用户动态地向系统提交作业步，每完成一个作业步的动作，系统便给出相应的回答信息，用户继续提交下一个作业步，直至作业全部完成，用户和系统以交互方式工作。分时系统上的交互式作业比批处理系统上的作业在管理和控制方面要简单些。

分时系统的一个重要设计是分时地为所有终端用户服务，如在某终端用户输入命令的时候，处理机可能在处理其他终端用户已经输入的命令。为了保证交互的及时性，当分时系统的用户动态提交作业步时，系统必须立即响应并处理每个作业步的动作。对终端用户通过终端提交的作业步，系统必须加以识别并立即解释执行。为此，系统设有命令解释程序，由它解释终端所输入的命令。如图 1.8 所示是分时系统的运行环境。

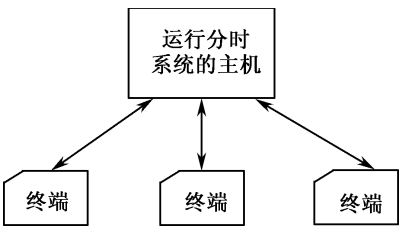

图 1.8　分时系统的运行环境

3．实时系统

随着计算机的不断普及和发展，计算机的应用领域日益扩大。20 世纪 60 年代后期，计算机已广泛应用于工业控制、军事控制及商业事务处理等领域。这类应用领域对计算机系统提出了新的要求，希望系统对来自外部的信息能在规定的时限内做出处理，我们称之为实时处理。

所谓“实时”（Real Time），可以理解为立即、及时的意思，是指计算机的运算和处理时间与被控过程或事务处理所需的真实时间相适应。我们把面向这类实时应用的计算机系统称为实时系统。虽然实时系统大都具有专用性，而且其种类、规模及对实时性的要求程度各不相同，但对于大、中型实时系统来说，绝大部分都以多道程序设计技术为基础，所以在资源管理、并发控制等方面与其他类型系统具有相同的基本特性。实时系统与其他类型系统的本质差别在于“及时性”，即实时系统应能及时地响应外部事件的请求并在严格规定的时间内完成对该事件的处理，控制实时设备和实时任务协调一致地运行。“高可靠性”也是实时系统的主要设计目标之一。为了提高实时系统的及时性和可靠性，软、硬件都必须采用相应的措施加以保证。

“实时”应用可分为两类。

① 实时控制

例如，将计算机用于飞行器的飞行自动控制属于实时控制应用。在这类应用中，计算机要对测量系统所测得的数据及时进行处理并及时输出，以便对被控目标进行及时控制或向控制人员显示结果。类似地，把计算机用于工业控制，如用计算机控制炼钢，这时计算机要对传感器定时送来的“炉温”数据及时进行处理，然后控制相应的机构使得炉温按照一定的规律变化或恒定不变。这类应用被称为“实时控制”。这类系统必须确保及时，因此又称为强实时系统。现

在诸多在各类控制系统的计算机中运行的“嵌入式操作系统”都属于实时控制类系统。

② 实时事务处理

例如，把计算机用于飞机订票系统、银行管理系统等属于实时事务处理应用。在这类应用中，计算机能对用户的服务请求及时做出回答，并能及时修改、处理系统中的数据。这类应用被称为“实时事务处理”。

实时系统的主要特征和功能如下。

① 时钟分辨度高。有更高的时钟中断频度，可实现更精确的计时，因此时钟分辨度高，可以更加频繁地进入操作系统“处理机（进程）调度程序”运行，保证实时任务及时占用处理机，以此保证实时任务的快速响应。

② 支持可剥夺任务调度。保证实时任务无条件地剥夺非实时任务运行，不会让非实时任务耽误实时任务。

③ 多级中断机制。保证实时任务对应的事件中断为高级。例如，在计算机控制的炼钢系统中，能对传感器“炉温”数据采集设备对应的中断及时进行处理，允许它的中断打断其他诸如键盘中断等低级中断的处理程序。

20 世纪 60 年代是操作系统不断成熟、蓬勃发展的重要时期，不仅先后出现了多道批处理系统、分时系统和实时系统，而且操作系统的基本理论、原理、基本技术和设计方法也已日趋成熟。从本节的介绍中读者不难看出，各种不同类型的系统均基于多道程序设计系统。故本书并未专辟章节具体介绍各类系统，而是以多道程序设计系统为核心介绍操作系统的基本理论、原理、基本技术和设计方法。同时在具体讨论各种资源管理的策略和方法时，对其所适用的环境也加以研究和分析。

1.2.3 多种方式操作系统

20 世纪 60 年代中期，随着计算机集成电路和操作系统的飞速发展，用户对计算机和操作系统的要求不断提高。新机种的不断出现，一方面满足了用户的需要，另一方面也给用户带来了某些不便，这是因为用户使用新机器势必要舍弃在老机器上已通过的程序。而随着应用广度与深度的不断扩展，原有程序已成为一笔巨大的财富。对此，IBM 公司首先推出了系列机概念。

所谓系列机就是在同一系列的机器中，新型号的机器能与老型号的机器兼容，这样用户既可以立即使用更高级的机器，又能在新机器上继续使用原有的程序，两全其美。为了满足用户的需要，计算机也被设计成容量大、功能全、几乎能提供用户需要的所有功能的通用计算机。与这种形势相适应，第三代操作系统被设计成多种方式操作系统，即一个操作系统既能处理批量作业，也能处理分时、实时等作业。这类系统的典型代表是 UNIX、VMS（DEC VAX 机器上的操作系统）操作系统。

多种方式操作系统不仅给用户提供了很大方便，而且对计算机资源的利用也更为合理。在单方式操作系统中，可供运行的作业类型受到限制，而多种方式操作系统能处理任何类型的作业，可将各种类型的作业合理搭配，系统更容易达到饱和状态，从而更加有利于提高资源的利用率。

1.2.4 个人计算机操作系统、并行与分布式操作系统及其发展

20 世纪 80 年代初，计算机已十分普及，特别是个人计算机（Personal Computer，PC）已开始进入千家万户，使得人们对计算机的使用要求更高了。这时，人们不仅仅满足于在计算中心使用计算机，并且要求在办公室、家中也能方便地使用计算机并能够交换信息。这促进了计算机网络的发展，人们把许多计算机连成网络。在网络中的计算机可以相互通信，共享软、硬件

资源，从而可以让用户通过多种多样的终端或工作站访问遍布各地的计算机网络。计算机网络的出现给计算机的发展和应用带来了动力，同时也给操作系统提出了新的问题。

为了能控制计算机网络，人们提出了“分布式操作系统”（Distributed Operating System）的理想境界。在为分布式操作系统理想而奋斗的实践中，网络操作系统派生出来。1983 年诞生的 4.2BSD UNIX 包含了对 TCP/IP 通信协议的支持，它是网络操作系统的代表。分布式操作系统与传统的操作系统有很多共同之处，但是它面临着传统操作系统所没有遇到过的网络中机间通信的问题。分布式操作系统在传统操作系统的基础上增加了许多机器定位、机间通信等方面的内容。

20 世纪 90 年代以来，随着共享主存的对称多处理机系统的广泛应用，“多机操作系统”也已经成熟。多机操作系统不同于单机操作系统，它支持多个处理机真正地并行运行多个程序。多机操作系统以支持并行多任务为其主要特征，充分发挥计算机中多处理机并行处理的优势，在科学计算及高端事务处理服务器领域占有重要地位。

1985 年，微软公司受苹果公司的 Macintosh 操作系统窗口式人机界面的影响，发布了基于 DOS 的 Windows 操作系统。微软公司的“Windows 系列操作系统”，以其特有的图形化人机界面，为计算机被普通用户所接受起到了决定性作用。这种操作系统以图形显示设备、鼠标、键盘等 I/O 外部设备为基础，配备有相应图形显示驱动程序及窗口管理模块。各种实用程序不再用文本行的界面，取而代之的是图形化的窗口界面，这使计算机更加接近大众，更加快速地得到普及。

操作系统自形成以来，经过几十年的发展，其基本原理和设计方法已趋于成熟。出现了许多得到广泛公认的流行系统，如 UNIX，Windows 操作系统等。20 世纪 80 年代之后，随着通用微处理器芯片的高速发展，个人计算机和工作站系统得到了迅猛发展，强烈冲击着传统小型机和中大型机市场。相应地，微机及工作站上的操作系统获得了快速的发展和应用（如 Windows，Solaris，IRIX 等）。从操作系统的发展历史看，推动其发展的动力主要来自计算机系统的不断创新和计算机应用的不断深入。

当前操作系统还在不断发展过程中，以下 4 个方面是发展的研究重点。

① 嵌入式操作系统的研究。伴随着智能手机、机器人、智能家电家具等设备的发展，对操作系统在功能和所占储存空间大小权衡方面，对实时响应和交互界面，对设备间通信都提出了新的要求。

② 强实时操作系统的研究。对操作系统的实时响应要求从来没有停止过，要求计算机的最大响应时间越来越短，对任务调度时机、算法的要求越来越高，特别是针对通用操作系统的实时性研究还在不断发展当中。

③ 并行操作系统的研究。随着高性能通用微处理器的发展，人们已经成功地提出了用它们构造“多处理机并行”的体系结构，如基于共享主存的对称多处理机系统（SMP）、多核系统。用成千上万个微处理器实现基于分布式存储的大规模并行处理机系统（MPP），这类被称为巨型机的并行系统有着良好的发展前景。建立在这类并行机上的操作系统与传统操作系统有明显的区别，其突出特征是提供各类并行机制，如并行文件系统、并行 I/O 控制、多处理机及协处理机的分配和调度、处理机间的通信和同步、用户任务的并行控制等。

④ 网络操作系统和分布式操作系统的研究。计算机网络系统和分布式操作系统已经深入人心。就目前情形而言，网络操作系统也还在不断进化当中，基于 Client/Server 模型的分布式操作系统也已不断走向应用，完全分布式的操作系统还未成形，仍将是研究的热点问题。另外，集群计算结构及网格结构的发展、新型网络存储的发展，给操作系统及其文件系统研究带来了新的挑战。

1.3 主要操作系统介绍

目前最常用的操作系统是 Windows 系列、UNIX 家族和 Linux。其中，UNIX 常用的变种有 SUN 公司的 Solaris、IBM 公司的 AIX、惠普公司的 HP UX 等。Linux 内核的实现思想也借鉴了 UNIX，而其命令界面及系统调用接口几乎和 UNIX 一样。其他比较常用的操作系统还有 Mac OS，NetWare 等。

1.3.1 Windows 系列及 MS DOS

微软公司于 1975 年成立，最初只有一个 BASIC 编译程序和比尔·盖茨、保罗·艾伦两个人。但现在，微软公司已成为世界上最大的软件公司，其产品涵盖操作系统、开发系统、数据库管理系统、办公室自动化软件、网络应用软件等各个领域。Windows 系列是由微软公司从 1985 年起开发的一系列操作系统产品，包括个人（家用）、商用和嵌入式三条产品线。Windows 系列操作系统主要产品如图 1.9 所示。

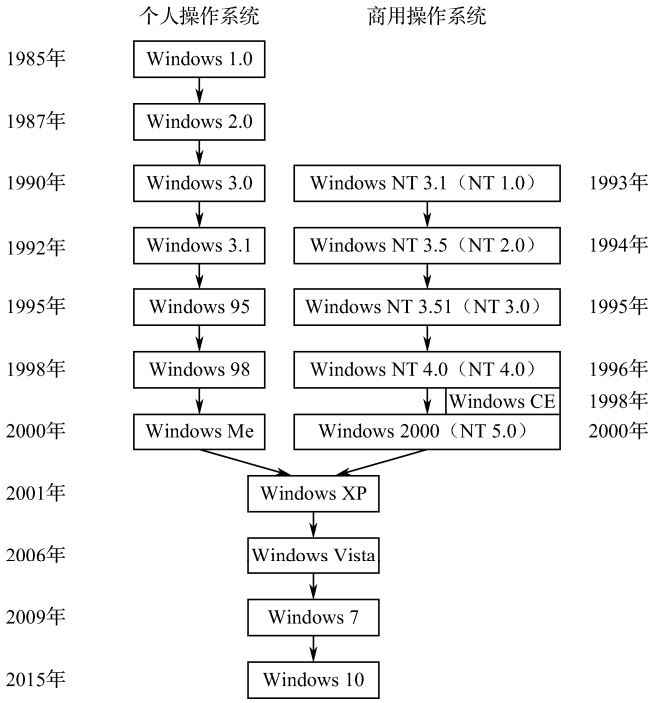

图 1.9　Windows 系列操作系统主要产品

微软公司从 1983 年开始研发 Windows 操作系统。当时，IBM PC 已进入市场，微软公司开发的个人计算机操作系统 DOS 和编程语言 BASIC 的编译器随 IBM PC 捆绑销售，取得了成功。Windows 操作系统最初的研制目标是在 DOS 基础上提供一个多任务的图形用户界面。不过，第一个取得成功的图形用户界面操作系统并不是 Windows 操作系统，而是 Windows 操作系统的模仿对象—— 苹果公司于 1984 年推出的 Mac OS（运行于 Macintosh 机上）。Macintosh 机及其上的 Mac OS 风靡美国多年，是当时 IBM PC 和 DOS 操作系统在市场上的主要竞争对手。当年，苹果公司曾对 Windows 操作系统不屑一顾，并大力抨击微软公司抄袭 Mac OS 的外观和灵感。但 Macintosh 机和 Mac OS 是封闭式体系（硬件接口不公开、系统源代码不公开等），而 IBM PC 和

MS DOS 是开放式体系（硬件接口公开、允许并支持第三方厂家制造兼容机、操作系统源代码公开等），这个关键的区别使得 IBM PC 后来者居上，销量超过了 Macintosh 机，并使在 IBM PC 上运行的 Windows 操作系统的普及率超过了 Mac OS，成为在个人计算机市场占主导地位的操作系统。

1．MS DOS

DOS 是微软公司与 IBM 公司开发的、广泛运行于 IBM PC 及其兼容机上的操作系统，全称是 MS DOS。

20 世纪 80 年代初，IBM 公司开发了 IBM PC。当其涉足个人计算机市场时，曾多方考察选择配合该机的操作系统。1980 年 11 月，IBM 和微软公司正式签约，日后的 IBM PC 均使用 DOS 作为标准的操作系统。由于 IBM PC 大获成功，微软公司也随之得到了飞速发展，MS DOS 从此成为个人计算机操作系统的代名词，发展为个人计算机的标准平台。

IBM PC 上所配的操作系统称为 PC DOS，是 IBM 公司向微软公司买下 MS DOS 的版权，经过修改和扩充后而产生的。

MS DOS 最早的版本是 1981 年 8 月推出的 1.0 版，1993 年 6 月推出了 6.0 版，最后一个 MS DOS 版本是 DOS 6.22，以后不再有新的版本。MS DOS 是一个单用户个人计算机操作系统，自 4.0 版开始具有多任务处理功能。

2．Windows 3.1，Windows 95/98 及 Windows Me 的发展历史

微软公司 Windows 操作系统的个人产品线由 20 世纪 80 年代的 DOS 演变而来，其中，影响较大和较突出的版本是 Windows 3.1 和 Windows 95。

Windows 3.1 在 1992 年发布，该系统修改了 3.0 版的一些不足，并提供了更完善的多媒体功能。Windows 操作系统开始流行起来，确定了其在 PC 领域的垄断地位。而 Windows 95 一上市就风靡世界。Windows 3.1 及以前的版本均为 16 位系统，因而还不能充分利用硬件的发展提供强大功能。同时，它们必须与 DOS 共同管理系统硬件资源，依赖 DOS 管理文件系统，且只能在 DOS 之上运行，并且它不是多道程序设计系统，因而还不能算是完整的操作系统。而 Windows 95 则重写了操作系统内核，不再基于 DOS 系统，特别是增加了多任务，使得用户可以同时运行多个程序，成了真正的多道程序设计系统，并在提供强大功能（如网络和多媒体功能等）和简化用户操作（如桌面和资源管理等新特性）两个方面都取得了突出的成绩。

2000 年 9 月，微软公司推出 Windows 95/98 的后续版本 Windows Me（视窗千禧版，Microsoft Windows Millennium Edition），较之 Windows 95/98 没有本质上的改进，只是扩展了一些功能，增加了一些驱动程序。Windows Me 的后续版本是把微软公司个人产品线与商用产品线合二为一（即把 Windows Me 和 Windows 2000 合二为一）的 Windows XP。这种产品线的合并同时意味着，微软公司以后的个人和商用机器的 Windows 操作系统都基于 Windows NT 内核。

3．微软公司的多用户操作系统 Windows NT 系列

微软公司在 20 世纪 80 年代中后期的主流产品 Windows 和 DOS 都是个人计算机上的单用户操作系统。1985 年，IBM 公司开始与微软公司合作开发商用多用户操作系统 OS/2，1987 年 OS/2 推出后，微软公司开始计划建立自己的商用多用户操作系统。1988 年 10 月，微软公司聘用 Dave Culter 作为 Windows NT 的主设计师，开始组建开发新操作系统的队伍。1993 年 5 月 24 日，经过几百人 4 年多的工作，微软公司正式推出 Windows NT。在相继推出 Windows NT 1.0，2.0，

3.0，4.0 后，2000 年 2 月，推出 Windows 2000（原来称为 Windows NT 5.0）。而 Windows 2000 的后一个版本是 Windows XP。后来又推出了 Windows 2003。2006 年推出 Windows Vista，2009 年推出 Windows 7，系统性能得到很大改善。2015 年又推出了 Windows 10。

Windows NT 的目标是开发在工作站和服务器上使用的 32 位操作系统，以充分利用 32 位微处理器等硬件新特性，并使其容易适应将来的硬件变化，同时不影响已有应用程序的兼容性（使原有的工作量和修改量最小）。

Windows NT 设计初期采用 OS/2 界面，后来因 Windows 3.1 操作系统的成功又改用 Windows 系列的界面。Windows NT 系列可支持 Intel x86 和部分 RISC CPU。Windows NT 较好地实现了充分利用硬件新特性、可扩充性、可移植性、兼容性等设计目标，它支持对称多处理机结构、内核级多线程、多个可装卸文件系统（MS DOS FAT、OS/2 HPFS、CDROM CDFS、NT 可恢复文件系统等），还支持多种常用 API（应用编程接口，如 WIN 32，OS/2，DOS，POSIX 等），提供源码级兼容或二进制兼容，内置网络和分布式计算，具有互操作性。Windows NT 的安全性达到了美国政府 C2 级安全标准。

Windows NT 的后一个版本与 Windows Me 的后一个版本合二为一，称为 Windows XP。Windows XP 的设计理念是，把以往 Windows 系列软件个人版的易用性和商用版的稳定性集于一身。

4．嵌入式操作系统

微软公司推出的嵌入式操作系统有用于掌上电脑的 Windows CE 和用于手机的 Smartphone 等。

1.3.2 UNIX 大家族

UNIX 是一种多用户操作系统。其在 1969 年诞生于美国贝尔实验室。由于其具有简洁、易于移植等特点而很快得到注意、发展和普及，成为从微型计算机跨越到巨型计算机的唯一操作系统。除贝尔实验室的“正宗”UNIX 版本外，UNIX 还有大量的变种。目前的主要变种有 SUN Solaris，IBM AIX 和 HP UX 等，不同变种间的功能、接口、内部结构与函数基本一致但又有不同。除变种外，还有一些系统，如 Mach 和 Linux，它们与变种的 UNIX 不同。变种是在正宗版本的基础上修改（包括界面与内部实现）而来的，而 Mach 和 Linux 只是系统调用接口相同，其内核则完全被重写。

最早的 UNIX 具有内核结构小巧、接口简单统一、功能丰富实用、用高级语言编写、可移植性好、源代码免费开放等优点，这些优点对 UNIX 的迅速成功起着重要作用。但后来由于变种不加控制地繁衍，以及功能的不断增加，其中的一些优点并没有完全保持下来。后来的 UNIX 内核结构结构不再小巧，而变得越来越庞大、复杂和笨拙。源代码免费开放和简单的许可证传播形式促进了早期的普及，但也导致了后来各变种间的不兼容。

此外，最初的 UNIX 就有内核结构可扩充性不强、缺乏图形界面、接口对初学者和普通用户不友好等缺点。现在这些缺点有的得到了改善，有的还存在。图形界面在后来得到了发展，如 X-Window，Motif 等。但是，内核结构问题至今仍存在。

UNIX 最初的许多概念、命令、实用程序和语言，今天仍沿用，显示了 UNIX 原始设计的简洁和高效。

UNIX 的发展主要经历了 5 个阶段。其发展历程如图 1.10 所示。

（1）UNIX 的产生

UNIX 起源于 Multics 项目开发的失败。Multics（Multiplexed Information and Computing Service 的简称）项目由 AT&T 公司的贝尔实验室、通用电气（General Electric）公司和麻省理工

学院联合开发，旨在建立一个能够同时支持数千用户的分时系统。该项目因目标过于庞大而失败，于 1969 年撤销。

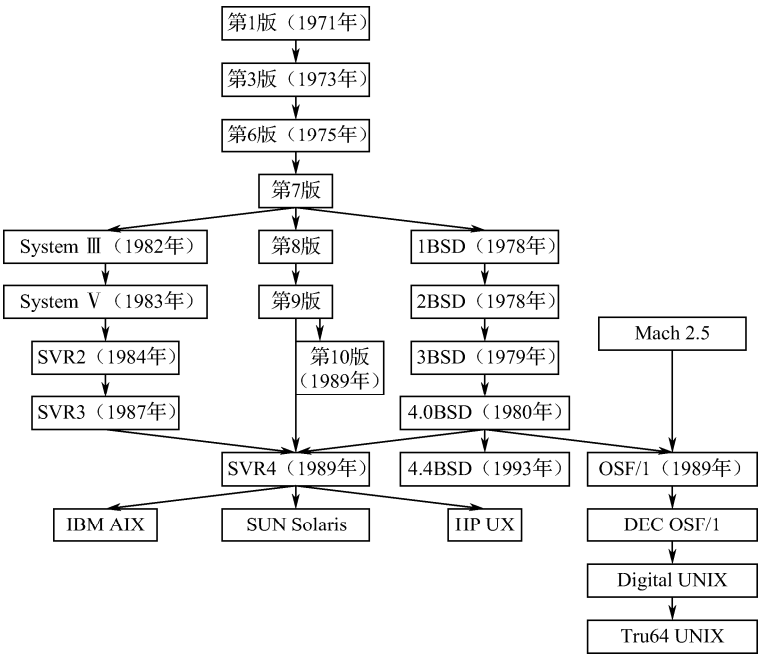

图 1.10　UNIX 发展历程

退出 Multics 项目后，贝尔实验室的雇员 Thompson 开始在公司的一台闲置的 PDP-7 机（主存 4KB）上开发一个“太空漫游”游戏程序。由于 PDP-7 机缺少程序开发环境，为了方便这个游戏程序的开发，Thompson 和公司的另一名雇员 Ritchie 一起用 GE-645 汇编语言（以前用于 Multics 项目开发）开发 PDP-7 机上的操作环境。最初是一个简单的文件系统（后来演化为 S5 文件系统），很快又添加了一个进程子系统、一个命令解释器（后来发展为 Bourne Shell）和一些实用工具程序，这个系统被命名为 UNIX。

此后，随着贝尔实验室工作环境的需要，UNIX 被移植到 PDP-11 机上，并逐渐增加了新的功能。很快地，UNIX 开始在贝尔实验室内部流行，许多人都加入对它的开发。1971 年，《UNIX 程序员手册》第 1 版出版。从这之后直到 1989 年，贝尔实验室又相继推出 10 个版本的 UNIX。

1973 年，Ritchie 开发的 C 语言对 UNIX 的发展起了重要作用。同年，用 C 语言重写了 UNIX（UNIX 第 4 版），这使得 UNIX 的可移植性大大增强，这是 UNIX 迈向成功的关键一步。

1973 年 10 月，Thompson 和 Ritchie 在 ACM（Association for Computing Machinery，计算机协会）的 SOSP（Symposium on Operating Systems Principles，操作系统原理讨论会）上发表了首篇 UNIX 论文，这是 UNIX 首次在贝尔实验室以外亮相。

（2）免费扩散（1973 年到 20 世纪 70 年代末）

自从在 SOSP 上发表论文后，UNIX 马上引起了众人的注意和兴趣。UNIX 软件和源代码迅速以许可证形式免费传播到世界各地的大学及研究机构中。这些大学、研究机构在免费使用的同时，对 UNIX 进行了进一步的研究、改进和移植。贝尔实验室又将这些改进与移植加到其以后的 UNIX 版本中。这种管理员与用户之间的敬业精神正是 UNIX 快速成长和不断发展的关键因素。UNIX 早期发展的这种特征与近年 Linux 的发展极为相似。

另外，众多大学对 UNIX 的免费使用，使学生得以熟悉 UNIX。这些学生毕业后又把 UNIX

传播到各种商业和政府机构中，这对 UNIX 早期的传播和普及起到了重要作用。

UNIX 的第一次移植是由 Wollongong 大学于 1976 年在 Interdata 机上进行的。其他几次较早的移植包括：1978 年，微软公司与 SCO 公司合作将 UNIX 移植到 Intel 8086 上，即 XENIX 系统（最早的 UNIX 商业变种之一）；1978 年，DEC 公司将 UNIX 委托移植到 VAX 上，即 UNIX/32V（3BSD 的前身）。

（3）商用版本的出现和三大主线的形成（20 世纪 70 年代中期到 80 年代中期）

20 世纪 70 年代中期到 80 年代中期，众多大学和公司的参与，使 UNIX 的变种迅速增多。像 SUN Solaris，IBM AIX 和 HP UX 等变种主要基于以下三条主线上的某个版本发展而来：由贝尔实验室发布的 UNIX 研究版（UNIX 第 1 版到 UNIX 第 10 版，或称 V1 到 V10，以后不再发行新版），由加州大学伯克利分校发布的 BSD（Berkeley Software Distribution），由贝尔实验室发布的 System Ⅲ 和 System Ⅴ。

1984 年的 AT&T 公司大分家使得 AT&T 公司可以进入计算机市场。因此，除贝尔实验室研究小组继续研究和发行 UNIX 研究版外，AT&T 公司成立了专门的 UNIX 对外发行机构。这些机构先后发行了 System Ⅲ（1982 年）、System Ⅴ（1983 年）、System Ⅴ Release 2（SVR2，1984 年）、System Ⅴ Release 3（SVR3，1987 年）。许多商业 UNIX 变种都是基于这条主线实现的。

加州大学伯克利分校是最早（1973 年 12 月）领取许可证的 UNIX 用户之一。最初的 BSD 版本发行（1978 年春的 1BSD 和 1978 年末的 2BSD）仅包含应用程序和实用工具（如 vi 解释器，Pascal 编译，C 编译，shell 解释器等），没有对操作系统内核本身进行修改和再发行。1979 年末的 3BSD 则在操作系统内核设计上实现了页式虚存，它是由加州大学伯克利分校发行的第一个操作系统内核。在 3BSD 中所做的虚存工作使该校得到美国国防部资助，推出了 4BSD（1980 年的 4.0BSD 到 1993 年的 4.4BSD），其中集成了 TCP/IP，引入了快速文件系统（Fast File System，FFS）、套接字等大量先进技术。BSD 对 UNIX 的发展具有重要影响，有许多新技术是 BSD 率先引入的。SUN OS 就是基于 4BSD 的。加州大学伯克利分校对 UNIX 的开发工作一直由计算机科学研究小组（Computer Science Research Group，CSRG）承担，1993 年发行 4.4BSD 时，CSRG 宣布因缺少资金等原因而停止 UNIX 开发，因此 4.4BSD 是加州大学伯克利分校发行的最后一个版本。

到 20 世纪 80 年代，UNIX 已在从微型计算机到巨型计算机等众多不同机型上运行。作为通用操作系统，当时 UNIX 的主要竞争对手是各计算机厂商的专有系统，如 IBM 的 OS 360/370 系列等。

（4）两大阵营和标准化（20 世纪 80 年代后期）

20 世纪 80 年代后期，UNIX 已经出现了很多变种。变种增多导致了程序的不兼容性和不可移植（同一应用程序在不同 UNIX 变种上不能不经修改而直接运行）。因此，迫切需要对 UNIX 进行统一标准化，这就导致了竞争的两大阵营和中间标准机构的出现。

1987 年，在统一市场的浪潮中，AT&T 公司宣布了与 SUN 公司的一项合作，将 System Ⅴ和 SUN OS 统一为一个系统。其余厂商（IBM，Digital，HP，Apollo 等）十分关注这项合作，认为他们的市场处于威胁之下，于是联合开发新的开放系统。他们的新机构称为开放软件基金会（Open Software Foundation，OSF），于 1988 年成立。作为回应，AT&T 和 SUN 公司联盟也于 1988 年形成了 UNIX International（UNIX 国际，UI）。以 SVR4 为契机的这场“UNIX 战争”将系统厂商划分成 UI 和 OSF 两大阵营，并围绕着两大主要 UNIX 系统技术（SVR4 和 OSF/1）展开竞争。

1989 年，在 SVR3，BSD 和 XENIX 的基础上，AT&T 的 UNIX Software Operation（UNIX

软件工作室，USO）设计实现了 SVR4。SVR4 是非常成功且广泛使用的一个版本（目前大部分 UNIX 商业变种都基于 SVR4）。SVR4 的开发对操作系统内核进行了大幅度重写，吸收了由 SUN OS 提供的许多特性，如虚拟文件系统（Virtual File System，VFS）接口、一个完全不同的存储管理体系结构和 SUN 网络文件系统（Network File System，NFS）。除从 SUN OS 吸收增强特性外，还从 BSD，SVR3 和 XENIX 中吸收了诸多特性，并对 UI 定义的新特性进行了补充。

定义 SVR4 的过程在当时来说是非常开放的。最终用户、软件开发商、系统管理员、经销商等都被要求列出他们的系统软件问题，并就他们对下一代 UNIX 环境的希望提供反馈。最终的目标是，确定如何最好地增强 UNIX 的可用性和可伸缩性。基于这些意见和反馈，SVR4 从当时的三个主要 UNIX 分支（BSD/SUN OS，SVR3，XENIX）汲取了最优的技术。

SVR4 通过把 UNIX 的三个主要分支联合到一个公用环境中来统一零碎的 UNIX 市场。SVR4 使已安装的 80%的 UNIX 系统得到了统一，使主要基于 SVR3、XENIX 的变种和基于 BSD/SUN OS 的变种得到了统一。SVR4 规范了系统调用等接口，使用户二进制程序在处理机型号相同的情况下跨操作系统版本移植成为可能。

与 UI 相对立的 OSF，则于 1989 年推出基于卡内基-梅隆大学 Mach 2.5 版的操作系统 OSF/1。

20 世纪 80 年代中期，由卡内基-梅隆大学开发的 Mach 支持 UNIX 系统调用编程接口，但却是一个全新的操作系统内核。OSF/1 和 NextStep 等商业系统都基于 Mach 2.5。Mach 2.5 的后续版本 Mach 3.0 与 2.5 版的内核结构完全不同，引入了所谓微内核结构。不过由于微内核结构的系统开销太大，因此一直没有商业化。

除 UI 和 OSF 所做的统一和标准化努力外，还出现了若干 UNIX 标准接口（主要是编程接口），如 AT&T 公司的 SystemⅤ接口定义（SystemⅤ Interface Definition，SVID）、IEEE POSIX 规范（1990 年的 POSIX 1003.1）、X/Open 可移植性指南（X/Open Portability Guideline，XPGA，如 1993 年的 XPG4）等。

任一标准都只涉及大多数 UNIX 系统功能的一个子集。从理论上说，如果程序员只使用该子集的函数，其应用程序就可以移植到任何遵从同一标准的系统中。但这也意味着，程序员不能利用硬件或操作系统特性对应用程序进行优化，也不可能利用某些 UNIX 厂商提供的特殊功能。标准化保障了可移植性，却给性能优化制造了障碍。与这一矛盾相关的是，在标准化过程中，各厂商总想加入一些特性来标榜自己的“产品特色和优势”，这也使得标准化没有完全成功。

（5）共同面对 Windows NT 的竞争和两大阵营的淡化（20 世纪 90 年代）

20 世纪 90 年代初期，美国经济萧条，加上微软公司的 Windows 操作系统蓬勃发展，这一切都威胁着 UNIX 操作系统的发展乃至生存。共同面对的外来竞争，使两大阵营（UI 与 OSF）的争斗很快淡化下来。

1993 年，UI 停止商业运作。出于各种原因，SVR4 从 1989 年至今几经易主，先后曾属于 AT&T UNIX Software Operation（1989 年），UNIX 系统实验室（UNIX System Laboratories，USL，1991 年），Novell 公司（1991 年部分股权，1993 年所有股权），X/Open（1993 年底，仅拥有商标和授权书）和 SCO 公司（1995 年底）。

Digital 公司于 1993 年发行的 DEC OSF/1 是唯一一个基于 OSF/1 标准的商业操作系统。此后，Digital 公司从该操作系统中删减了许多与 OSF/1 相关的部分，1995 年将其改名为 Digital UNIX。1998 年，Digital 公司被 Compaq 公司并购后又改名为 Tru64 UNIX。

1.3.3 自由软件 Linux 和 freeBSD 等

1984 年，自由软件的积极倡导者 Richard Stallman 组织开发了一个自由软件的软件体系——GNU（GNU is Not UNIX 的简称），并拟订了一份通用公用版权协议（General Public License，

GPL）。所谓自由软件，是指由开发者提供软件的全部源代码，任何用户都有权使用、复制、扩散、修改该软件，同时用户也有义务将自己修改的源代码公开。自由软件可免费提供给任何用户使用，也包括用于商业目的；并且自由软件的所有源代码也是公开的，可免费得到。它的源代码不仅公开而且可自由修改，无论修改的目的是使自由软件更加完善，还是在修改的基础上开发上层软件都没有问题。

自由软件的出现给人们带来很大的好处。首先，免费可使用户节省相当一笔费用；其次，公开源代码可吸引尽可能多的开发者参与软件的查错与改进。在开发协调人的控制下，自由软件新版本的公布、反馈、更新等过程是完全开放的。

目前，人们已很熟悉的一些软件，如因特网域名服务器程序 BIND、Perl 编程语言环境、Web 服务器程序 Apache、TCP/IP 网络软件等，实际上都是自由软件的经典之作。还有 C++编译器、FORTRAN 77、SLIP/PPP、IP Accounting、防火墙、Java 内核支持、BSD 邮件发送、Lynx 浏览器、Samba（用于在不同操作系统间共享文件和打印机）、Applixware 的办公套装、starOffice 套件、Corel Wordperfect 等都是著名的自由软件。

1993 年，Linux 的创始人 Linus Torvalds 把 Linux 奉献给了 GNU，从而使用户能够使用从底层到应用的全套自由软件。

1．Linux

Linux 是一个多用户操作系统，是由 Linus Torvalds 主持开发的遵循 POSIX 标准的操作系统。它提供了 UNIX 的编程界面，但内核实现则完全重写。虽然 Linux 内核是重新写的，但是其实现方法及数据结构基本上都借鉴了 UNIX 的思想。它是一个免费的自由软件，源代码开放，这是它与 UNIX 及其变种的不同之处。UNIX 虽然有过免费发送给学校和科研机构的时代，但是其所有者 AT&T 公司可以经营计算机产品后，开始收取转让费和使用费。由于内核可以转让，所以其变种也非常多，影响了对用户程序的兼容性。

Linus 在 2001 年初的 Linux World 大会前夕推出了 Linux 2.4 内核。Linux 有内核（Kernel）与发行套件（Distribution）两种版本号。内核版本指在 Linus 领导下的开发小组开发的系统内核的版本，如 Linux 2.6，即内核版本 2.6（一般来说，序号的第 2 位为偶数的版本表明这是一个可以使用的稳定版本，如 2.0.35；而序号的第 2 位为奇数的版本一般有一些新的东西加入，是一个不太稳定的测试版本，如 2.1.88）。一些组织机构或厂商将 Linux 内核同应用软件和文档包装起来，并提供一些安装界面、系统设定与管理工具，这样就构成了一个发行套件，如最常见的 Slackware，Red Hat，Debian，红旗 Linux 等。实际上，发行套件就是 Linux 的一个大软件包。相对于内核版本，发行套件的版本号随发布者的不同而不同，与内核的版本号是相对独立的，如 Slackware 3.5，Red Hat 9，Debian 1.3.1 等。

（1）Linux 的产生与发展

Linux 最初是由芬兰赫尔辛基大学计算机系的大学生 Linus Torvalds，在 1990 年年底到 1991 年的几个月中，为了自己的操作系统课程学习和后来上网使用而陆续编写的。他使用自己买的 Intel 386 PC，利用 Andrew S. Tanenbaum 设计的微型操作系统 Minix 作为开发平台。据 Linus 说，刚开始的时候他根本没有想到要编写一个操作系统内核，更没想到这一举动会在计算机界产生如此重大的影响。最开始是一个进程切换器，然后是为自己上网需要而自行编写的终端仿真程序，再后来是为他从网上下载文件而编写的硬盘驱动程序和文件系统。这时，他发现自己已经实现了一个几乎完整的操作系统内核。出于对这个内核的信心和发展愿望，Linus 希望这个内核能够免费扩散使用。但出于谨慎，他并没有在 Minix 新闻组中公布它，而只是于 1991 年底